BAB XII. CLASS

Class merupakan struktur data dari obyek. Untuk menjelaskan tentang class, lihat perbandingannya dengan struktur berikut ini:

Perhatikan program di bawah ini:

Contoh 1.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

struct mahasiswa
{
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

Setelah program di atas dicompile, error tidak ada.
Berikutnya struktur di atas kita ganti dengan class, menjadi

Contoh 2.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

class mahasiswa
{
      char nim[8]; char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

setelah program di atas di compile, ternyata error muncul. Error tersebut muncul karena class tidak dikenal dalam main(). Kesalahan ini sekaligus menunjukkan perbedaan dengan struktur.

Penggunaan PUBLIC

Agar program di atas dapat dicompile, ditambahkan perintah public diikuti dengan tanda titik dua (:), sehingga programnya menjadi

Contoh 3.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

class mahasiswa
{
   public:
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
      strcpy(mhsd3.nama, "Burhanudin Harahap");
      mhsd3.umur = 20;

      cout << mhsd3.nim << endl;
      cout << mhsd3.nama << endl;
      cout << mhsd3.umur << endl;
}
```

Perintah PUBLIC menyatakan bahwa perintah-perintah yang ada di bawahnya dapat diakses diluar class. Perintah PUBLIC merupakan termasuk a*ccess specifier* (penentu akses). Selain PUBLIC, terdapat perintah lain yang termasuk access specifier, yaitu PRIVATE.

Contoh 4.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

class mahasiswa
{
 private:
      char nim[8];
      char nama[20];
      int umur;
};

void main()
{
      mahasiswa mhsd3;
      strcpy(mhsd3.nim, "M0197001");
```

```
	strcpy(mhsd3.nama, "Burhanudin Harahap");
	mhsd3.umur = 20;

	cout << mhsd3.nim << endl;
	cout << mhsd3.nama << endl;
	cout << mhsd3.umur << endl;
}
```

Setelah dicompile, error yang sama dengan sebelumnya muncul yaitu class tidak dapat diakses di main().
Perintah PRIVATE memberi pengertian bahwa perintah yang ada dibawahnya hanya dapat diakses dalam class tersebut, yang dalam hal ini adalah class mahasiswa.

variabel nim, nama, dan umur dalam class mahasiswa disebut data anggota. Selain data anggota, kita juga dapat menambahkan fungsi anggota.

Contoh 5.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

class mahasiswa
{
	private :
		char nim[8];
		char nama[20];
		int umur;

	public :
	void inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS)
	{
		strcpy(nim, NIMMHS);
		strcpy(nama, NAMAMHS);
		umur = UMURMHS;
	}

	void tampilkan()
	{
		cout << nim << endl;
		cout << nama << endl;
		cout << umur << endl;
	}

};

void main()
{
	mahasiswa mhsd3;
	mhsd3.inisialisasi("M0197001", "Burhanudin Harahap", 20);
	mhsd3.tampilkan();
}
```

Pada program di atas, fungsi inisialisasi() dan tampilkan() merupakan fungsi anggota dari class mahasiswa. Keduanya dibuat public karena akan diakses dari luar class, sedangkan data anggotanya (nim, nama, umur) dibuat private.

Mungkin Anda berpikir mengapa program terakhir (contoh 5) terlalu panjang, padahal akan diperoleh hasil yang sama dengan program sebelumnya (contoh 1). Anda memang benar, tapi contoh 5 ini merupakan cara pemrograman berorientasi obyek (PBO).

Perhatikan program contoh 5, khususnya pada kedua fungsi anggota. Kedua fungsi tidak punya prototype. Kita juga dapat memberikan prototype yang merupakan cara kedua dalam penulisan.

Contoh 6.

```
#include <iostream.h>
#include <conio.h>
#include <string.h>

class mhs
{
  private :
      char nim[8];
      char nama[20];
      int umur;

  public :
      void inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS);
      void tampilkan();
};

void main()
{
      mhs mhsd3;
      mhsd3.inisialisasi("M0197001", "Burhanudin Harahap", 20);
      mhsd3.tampilkan();
}

void mhs::inisialisasi(char *NIMMHS, char *NAMAMHS, int UMURMHS)
{
      strcpy(nim, NIMMHS);
      strcpy(nama, NAMAMHS);
      umur = UMURMHS;
}

void mhs::tampilkan()
{
      cout << nim << endl;
      cout << nama << endl;
      cout << umur << endl;
}
```

Cara kedua inilah yang sering dipilih oleh para programmer C++.

Berikut ini contoh-contoh program yang memanfaatkan class

Contoh 7.
Program untuk menyimpan data n data mahasiswa kemudian menampilkannya.

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
```

```
class mahasiswa
{
   public:
     char nim[20];
     char nama[50];
     int  umur;

     void tampilkan(char *NIMMHS, char *NAMAMHS, int UMURMHS)
          {
               cout << "NIM MHS  : " << NIMMHS << endl;
               cout << "NAMA MHS : " << NAMAMHS << endl;
               cout << "UMUR     : " << UMURMHS << endl;
          }
};

void main()
{
   mahasiswa mhsd3[50]; { tipe data array }
   char temp[10]; int n, i;
   clrscr();
   cout << "<< ENTRI DATA MAHASISWA D3 " << endl;
   cout << endl;
   cout << "Jumlah mahasiswa : ";
   cin.getline(temp, sizeof(temp));
   n = atoi(temp);
   for (i=0;i<=n-1;i++)
   {
     cout << "DATA - " << i+1 << endl;
     cout << "NIM MAHASISWA  : " ;
     cin.getline(mhsd3[i].nim, sizeof(mhsd3[i].nim));
     cout << "NAMA MAHASISWA : " ;
     cin.getline(mhsd3[i].nama, sizeof(mhsd3[i].nama));
     cout << "UMUR           : ";
     cin.getline(temp, sizeof(temp));
     mhsd3[i].umur = atoi(temp);
     cout << endl;
   }

   // tampilkan semua data
   cout << "-------------------------" << endl;
   cout << "DATA YANG MASUK" << endl;
   cout << "-------------------------" << endl;
   for (i=0;i<=n-1;i++)
   {
     cout << "DATA MAHASISWA " << i+1 << endl;
     mhsd3[i].tampilkan(mhsd3[i].nim, mhsd3[i].nama, mhsd3[i].umur);
     cout << endl;
   }
   getch();
}
```

Contoh 8.
Program untuk menjumlahkan dan mengurangkan 2 buah bilangan kompleks.

```
#include <iostream.h>
#include <conio.h>

class kompleks
{
       private:
            float real;
            float imajiner;

       public:
            void tambah(float real1, float imajiner1, float
                 real2, float imajiner2)
                 {
                    real = real1 + real2;
                    imajiner = imajiner1 + imajiner2;
                 }

                  void kurangi(float real1, float imajiner1,
            float
                 real2, float imajiner2)
                 {
                    real = real1 - real2;
                    imajiner = imajiner1 - imajiner2;
            }

            void tampilkan()
            {
              cout << "Hasilnya adalah : " << real << "+ "
                   << imajiner << 'i' << endl;
            }
};

void main()
{
      clrscr(); kompleks bilkompleks;
      float el_real1, el_real2, el_imaj1, el_imaj2;
      cout << "Bilangan Kompleks pertama" << endl;
      cout << "Elemen real     : ";
      cin  >> el_real1;
      cout << "Elemen imajiner : ";
      cin  >> el_imaj1;
      cout << "Bilangan Kompleks kedua" << endl;
      cout << "Elemen real     : ";
      cin  >> el_real2;
      cout << "Elemen imajiner : ";
      cin  >> el_imaj2;
      bilkompleks.tambah(el_real1,el_imaj1,el_real2,el_imaj2);
      bilkompleks.tampilkan();

      bilkompleks.kurangi(el_real1,el_imaj1,el_real2,el_imaj2);
      bilkompleks.tampilkan();
      getch();
}
```